TENSOR DECOMPOSITION MODELS

Why do we want to factorize a matrix/tensor?

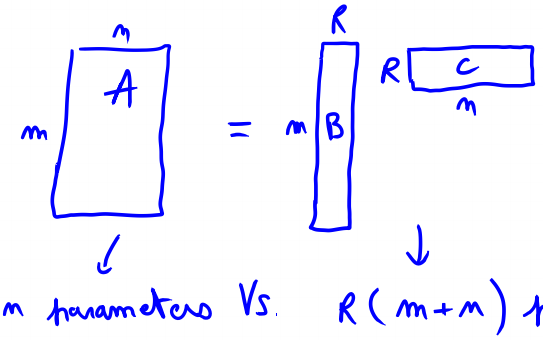
- ↳ Compression / reduce model size
- ↳ Discover structure in data (PCA)
- ↳ Infer missing entries (completion)

(I) MATRIX LOW RANK FACTORIZATION

Theorem: Let $A \in \mathbb{R}^{m \times n}$

$\text{rank}(A) \leq R \iff$ There exists $B \in \mathbb{R}^{m \times R}$ and $C \in \mathbb{R}^{R \times n}$ such that $A = BC$

↳ "rank R factorization of A"



mn parameters Vs. $R(m+n)$ parameters

Sometimes, an approximate factorization is enough:

$$A \approx BC$$
$$m \times n \quad m \times R \quad R \times n$$

Low rank approximation Problem: Given $A \in \mathbb{R}^{m \times n}$, and a target rank R

$$\min_{\substack{B \in \mathbb{R}^{m \times R} \\ C \in \mathbb{R}^{R \times n}}} \| A - BC \|_F$$

Def (Frobenius norm): $\| A \|_F = \sqrt{\sum_{i,j} (A_{ij})^2}$

Def (Singular Value Decomposition, SVD)

Any matrix $A \in \mathbb{R}^{m \times n}$ $(m \leq n)$ can be decomposed into

$$A = U D V^T \quad (1)$$

($m \times n$ = $m \times m$ $m \times m$ $m \times n$)

where U and V are orthonormal (i.e. $U^T U = I$ and $V^T V = I$) and D is a diagonal matrix with non-negative entries.

The columns of U are the left singular vectors: $u_1, u_2, \dots, u_m \in \mathbb{R}^m$

The columns of V are the right singular vectors: $v_1, v_2, \dots, v_m \in \mathbb{R}^m$

The diagonal entries of D are the singular values $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \dots \geq \sigma_m \geq 0$

Remark: • We can rewrite the SVD (1): $A = \sum_{i=1}^{m} \sigma_i u_i v_i^T$

• If $\text{rank}(A) = R$, then $\sigma_{R+1} = \sigma_{R+2} = \dots = \sigma_m = 0$ and $A = \sum_{i=1}^{R} \sigma_i u_i v_i^T$

Theorem (Eckart-Young)

Let $A \in \mathbb{R}^{m \times n}$ and let $A = U D V^T = \sum_{i=1}^{m} \sigma_i u_i v_i^T$ be its SVD.

Then the solution of

$$\min_{X \in \mathbb{R}^{m \times n}} \|A - X\|_F^2 \quad \text{subject to} \quad \text{rank}(X) \leq R$$

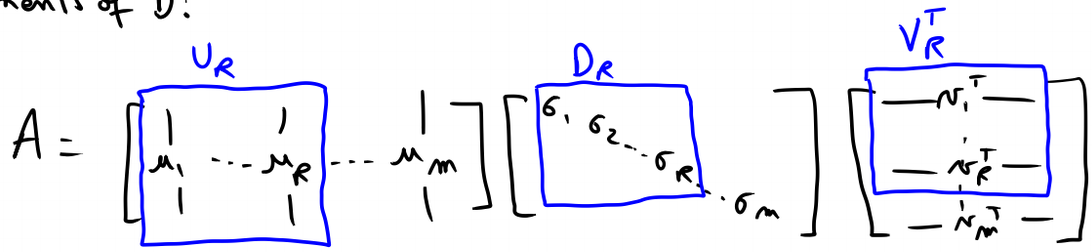
is given by $X^* = \sum_{i=1}^{R} \sigma_i u_i v_i^T$. (truncated SVD)

Low rank approximation problem: Given $A \in \mathbb{R}^{m \times n}$, and a target rank R

$$\min_{\substack{B \in \mathbb{R}^{m \times R} \\ C \in \mathbb{R}^{R \times n}}} \|A - BC\|_F$$

SOLUTION Compute SVD of $A = U D V^T$

keep the first R columns of U and V and the first R diagonal elements of D:



$$A \approx \underset{m \times R}{U_R} \underset{R \times R}{D_R} \underset{R \times n}{V_R^T} = \underbrace{(U_R D_R)}_{B,\ m \times R} \underbrace{(V_R^T)}_{C,\ R \times n}$$

(II) TENSOR NETWORKS (TN)

TN are graphs representing operations between tensors:

- Nodes represent tensors
- The arity (# of incoming edges / # of legs) of a node correspond to the order of the tensor:

$v \in \mathbb{R}^d$: v — d $\qquad$ $A \in \mathbb{R}^{m \times n}$: m — A — n

$T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$: T with legs d_1, d_2, d_3

- Edges represent contractions (summations):

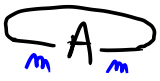
+ Two matrices: m — A — n $\qquad$ n — B — k

$$\left(\underset{m}{-}A\underset{n}{-}B\underset{k}{-} \right)_{i,j} = \sum_{k=1}^{n} A_{ik} B_{kj} = (AB)_{ij}$$

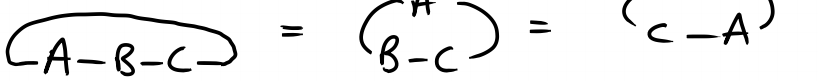
↪ $-A-B- = AB$

+ u — d — $v = \sum_{i=1}^{d} u_i v_i = \langle u, v \rangle$

+ $A \in \mathbb{R}^{m \times n}$, $v \in \mathbb{R}^n$ $\qquad \left(\underset{m}{-}A\underset{n}{-}v \right)_i = \sum_{j=1}^{n} A_{ij} v_j = (Av)_i$

+ $A \in \mathbb{R}^{m \times m}$ $\qquad$  $= \sum_{i=1}^{m} A_{ii} = \mathrm{Tr}(A)$

+ Proof of $\mathrm{Tr}(ABC) = \mathrm{Tr}(BCA) = \mathrm{Tr}(CAB)$ $\qquad (\neq \mathrm{Tr}(BAC))$

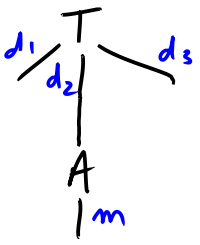


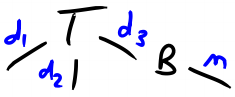
+ $u \in \mathbb{R}^m$, $v \in \mathbb{R}^n$ $\qquad \left(\underset{m}{u} \quad \underset{n}{v} \right)_{i,j} = u_i v_j = (u \otimes v)_{ij}$

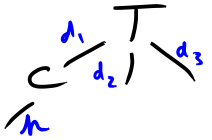
↪ u v $= u \otimes v$

$A \in \mathbb{R}^{m \times n}$, $B \in \mathbb{R}^{k \times q}$ $\qquad \left(\underset{m}{A}\ \underset{n}{\ } \quad \underset{k}{B}\ \underset{q}{\ } \right)_{ijke} = A_{ij} B_{ke}$

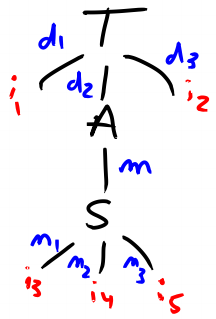
+ $T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$, $A \in \mathbb{R}^{m \times d_2}$, $B \in \mathbb{R}^{m \times d_3}$, $C \in \mathbb{R}^{n \times d_1}$

$\mathbb{R}^{d_1 \times m \times d_2} \ni$  $= T \times_2 A$

$\mathbb{R}^{d_1 \times d_2 \times m} \ni$  $= T \times_3 B$

$\mathbb{R}^{n \times d_2 \times d_3} \ni$  $= T \times_1 C$

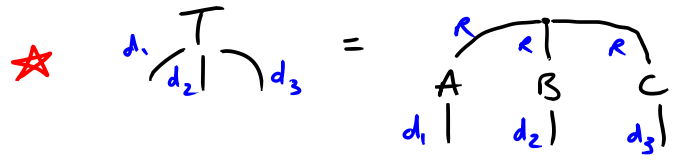
+ $T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$, $A \in \mathbb{R}^{d_2 \times m}$, $S \in \mathbb{R}^{m_1 \times m_2 \times m_3 \times m}$

 $= \sum_{j=1}^{d_2} \sum_{k=1}^{m} T_{i_1 j i_2} A_{jk} S_{i_3 i_4 i_5 k}$

+ $u, v, w \in \mathbb{R}^d$:  $= \sum_{i=1}^{d} u_i v_i w_i$

III 3 Tensor Decomposition Models

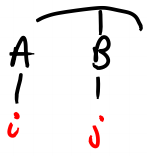
1) CP decomposition
↳ (CANDECOMP / PARAFAC)

★ 

← rank R CP decomposition of T

$A \in \mathbb{R}^{d_1 \times R}$, $B \in \mathbb{R}^{d_2 \times R}$, $C \in \mathbb{R}^{d_3 \times R}$

Def: The CP rank of a tensor T is the smallest R such that a rank R CP decomposition of T exists.

★ : $T_{ijk} = $ 

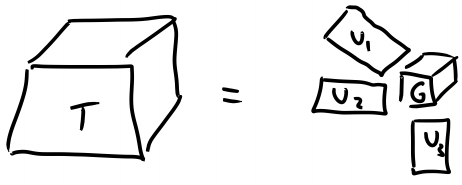
$= \sum_{r=1}^{R} A_{ir} B_{jr} C_{kr}$

If $A = \begin{pmatrix} | & & | \\ a_1 & \cdots & a_R \\ | & & | \end{pmatrix}$, $B = \begin{pmatrix} | & & | \\ b_1 & \cdots & b_R \\ | & & | \end{pmatrix}$, $C = \begin{pmatrix} | & & | \\ c_1 & \cdots & c_R \\ | & & | \end{pmatrix}$

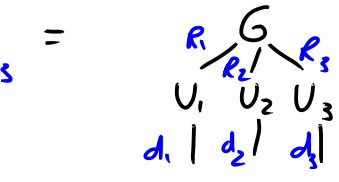
then $T_{ijk} = \sum_{r=1}^{R} (a_r)_i (b_r)_j (c_r)_k = \sum_{r=1}^{R} (a_r \circ b_r \circ c_r)_{ijk}$

↳ $T = \sum_{r=1}^{R} a_r \circ b_r \circ c_r$

2) Tucker decomposition



$T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$, $G \in \mathbb{R}^{R_1 \times R_2 \times R_3}$ ↳ "core tensor", $U_i = \mathbb{R}^{d_i \times R_i}$ for $i = 1,2,3$ ↳ "factor matrices"

(Δ) 

← rank (R_1, R_2, R_3) Tucker decomposition of T.

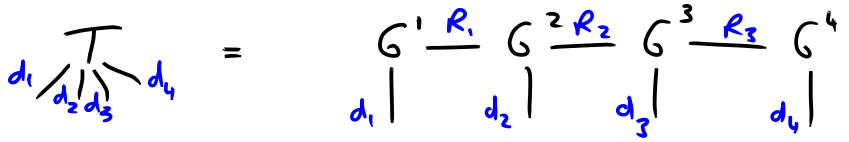
def: The multilinear rank (Tucker rank) of $T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ is the smallest tuple (R_1, R_2, R_3) s.t. a rank (R_1, R_2, R_3) Tucker decomposition of T exists.

↳ The Tucker decomposition (Δ) can be written as $T = G \times_1 U_1 \times_2 U_2 \times_3 U_3$

3) Tensor train decomposition (TT)

$T \in \mathbb{R}^{d_1 \times d_2 \times d_3 \times d_4}$, $\underbrace{G^1 \in \mathbb{R}^{d_1 \times R_1}, G^2 \in \mathbb{R}^{R_1 \times d_2 \times R_2}, G^3 \in \mathbb{R}^{R_2 \times d_3 \times R_3}, G^4 \in \mathbb{R}^{R_3 \times d_4}}$

"core tensors"



↳ rank (R_1, R_2, R_3) TT decomposition of T.

Def: The TT rank of a tensor $T \in \mathbb{R}^{d_1 \times d_2 \times d_3 \times d_4}$ is the smallest (R_1, R_2, R_3) such that rank (R_1, R_2, R_3) TT decomposition of T exists.

(IV) TENSOR LOW RANK APPROXIMATION

For $T \in \mathbb{R}^{d_1 \times d_2 \times \cdots \times d_N}$ ($d_1 d_2 \cdots d_N$ parameters)

	rank	# parameters	EXACT DECOMPOSITION	LOW RANK APPROXIMATION
CP	R	$R(d_1 + d_2 + \cdots + d_N)$	NP-hard	NP
TUCKER	$(R_1, R_2, \cdots, R_N)$	$R_1 R_2 \cdots R_N + \sum_{i=1}^{N} d_i R_i$	P (easy)	NP
TT	$(R_1, \cdots, R_{N-1})$	$d_1 R_1 + R_1 d_2 R_2 + R_2 d_3 R_3 + \cdots + R_{N-2} d_{N-1} R_{N-1} + R_{N-1} d_N$	P	NP

Low rank approximation problem :

1) CP Given $T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ and a target rank R :

$$\min_{\substack{A \in \mathbb{R}^{d_1 \times R} \\ B \in \mathbb{R}^{d_2 \times R} \\ C \in \mathbb{R}^{d_3 \times R}}} \left\| \underset{d_1 (d_2) d_3}{T} - \overset{R \quad R}{\underset{R}{\underset{d_1 \quad d_2 \quad d_3}{A \; B \; C}}} \right\|_F^2$$

Def : The Frobenius norm of $T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$: $\|T\|_F = \sqrt{\sum_{ijk} (T_{ijk})^2}$

2) TUCKER

Given $T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ and a target rank (R_1, R_2, R_3)

$$\min_{\substack{G \in \mathbb{R}^{R_1 \times R_2 \times R_3} \\ U_1 \in \mathbb{R}^{d_1 \times R_1} \\ U_2 \in \mathbb{R}^{d_2 \times R_2} \\ U_3 \in \mathbb{R}^{d_3 \times R_3}}} \| T - G \times_1 U_1 \times_2 U_2 \times_3 U_3 \|_F^2$$

3) TENSOR TRAIN

Given $T \in \mathbb{R}^{d_1 \times d_2 \times d_3}$ and a target rank (R_1, R_2)

$$\min_{\substack{G_1 \in \mathbb{R}^{d_1 \times R_1} \\ G_2 \in \mathbb{R}^{R_1 \times d_2 \times R_2} \\ G_3 \in \mathbb{R}^{R_2 \times d_3}}} \left\| \underset{d_1 (d_2) d_3}{T} - \underset{d_1 \quad d_2 \quad d_3}{G_1 \overset{R_1}{-} G_2 \overset{R_2}{-} G_3} \right\|_F^2$$

OPTIMIZATION WITH TENSORS

General problem: Given a loss function $\mathcal{L}: \mathbb{R}^{d_1 \times \cdots \times d_N} \to \mathbb{R}$, we want to solve

$$\min_{W \in \mathbb{R}^{d_1 \times \cdots \times d_N}} \mathcal{L}(W) \quad \text{subject to} \quad \operatorname{rank}(W) \leq R$$

↳ CP rank, Tucker rank, TT rank, ...

Examples of loss functions:

+ **Low rank approximation**

$$\min_W \|W - T\|_F^2 \quad \text{s.t.} \quad \operatorname{rank}_{CP}(W) \leq R$$

↳ the loss function is $\mathcal{L}(W) = \|W - T\|_F^2$

+ **Regression** We want to learn a linear function $f: \mathbb{R}^{d_1 \times \cdots \times d_N} \to \mathbb{R}$ from a dataset $\{(X_1, y_1), (X_2, y_2), \ldots, (X_m, y_m)\} \subseteq \mathbb{R}^{d_1 \times \cdots \times d_N} \times \mathbb{R}$.

f is linear $\Leftrightarrow f(X) = f_W(X) = \langle W, X \rangle = \sum_{i_1, \ldots, i_N} W_{i_1, \ldots, i_N} X_{i_1, \ldots, i_N}$.

In this case, a natural loss function is

$$\mathcal{L}(W) = \sum_{i=1}^m \left(f_W(X_i) - y_i \right)^2 = \sum_{i=1}^m \left(\langle W, X_i \rangle - y_i \right)^2$$

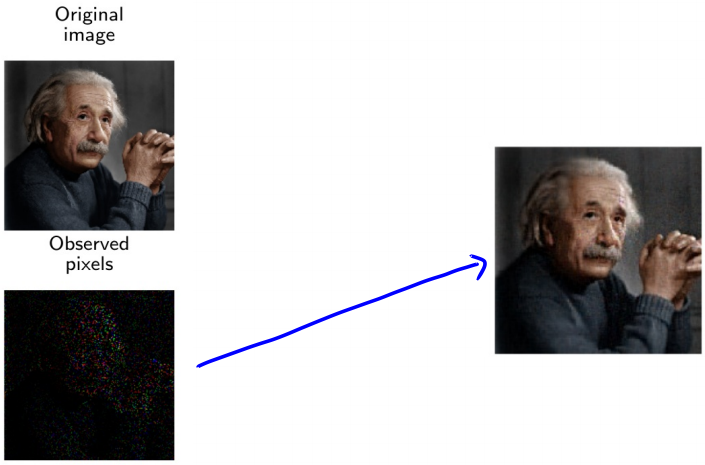
+ **Completion:** Target tensor $T \in \mathbb{R}^{d_1 \times \cdots \times d_N}$

Data: observed entries $\{T_{i_1, \ldots, i_N} \mid (i_1, \ldots, i_N) \in \Omega\}$

$\Omega \subseteq [d_1] \times [d_2] \times \cdots \times [d_N]$

- $\begin{pmatrix} 1 & ? & ? \\ ? & 2 & ? \\ ? & 3 & ? \end{pmatrix}$ $\quad \Omega = \{(1,1), (2,2), (2,3)\}$

- Image completion



For completion, the loss function is:

$$\mathcal{L}(W) = \sum_{(i_1,\dots,i_N)\in\Omega} \left(T_{i_1\cdots i_N} - W_{i_1\cdots i_N}\right)^2$$

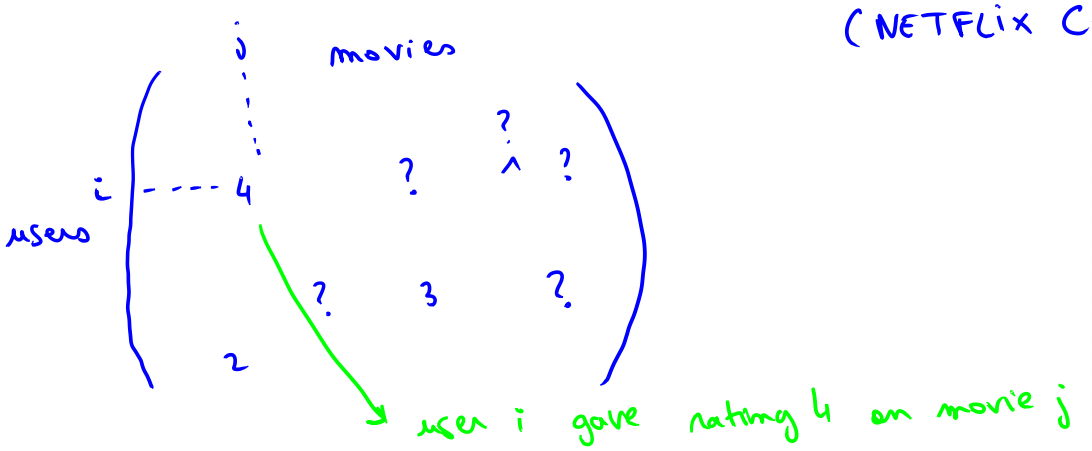
Ω ↳ set of observed entries

Remark: If $\Omega = [d_1]\times[d_2]\times\cdots\times[d_N]$, then

$$\sum_{(i_1,\dots,i_N)\in\Omega} \left(T_{i_1\cdots i_N} - W_{i_1\cdots i_N}\right)^2 = \|T-W\|_F^2$$

- COLLABORATIVE FILTER / RECOMMANDATION SYSTEMS

(NETFLIX CHALLENGE)

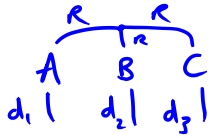


(I) General Optimization Algorithm

PBM 1 $\min_{W \in \mathbb{R}^{d_1 \times \dots \times d_N}} \mathcal{L}(W)$ subject to $\text{rank}_{CP}(W) \leq R$

Notation: Given $A_i \in \mathbb{R}^{d_i \times R}$ for $i = 1, \dots, N$, we denote the CP decomposition with factors $A_1, \dots, A_N$ by $CP(A_1, \dots, A_N)$

ex: $CP(A, B, C) =$



PBM 1 is equivalent to:

PBM 2 $\min_{\substack{A_i \in \mathbb{R}^{d_i \times R} \\ i=1,\dots,N}} \mathcal{L}\left(CP(A_1, \dots, A_N)\right)$

1) Gradient based algorithm

ALGORITHM: Initialize $A_1, A_2, \dots, A_N$

Repeat

  for $i = 1 \dots N$

    $X_i \leftarrow A_i - \gamma \nabla_{A_i} \mathcal{L}(CP(A_1, \dots, A_N))$ (γ: learning rate)

  for $i = 1 \dots N$

    $A_i \leftarrow X_i$

until convergence

2) Alternating minimization

We assume $\min_{A_i} \mathcal{L}(CP(A_1, \dots, A_N))$ is easy.

ALGORITHM: Initialize $A_1, A_2, \dots, A_N$

Repeat

  for $i = 1, \dots, N$

    $A_i \leftarrow \arg\min_{A_i} \mathcal{L}(CP(A_1, \dots, A_N))$

until convergence

3) Alternating minimization for low CP rank approximation

Given $T \in \mathbb{R}^{d_1 \times \cdots \times d_N}$, target rank R:

CP-PBM

$$\min_{\substack{A_i \in \mathbb{R}^{d_i \times R} \\ i=1,\dots,N}} \| T - CP(A_1, \dots, A_N) \|_F^2$$

- **Matricization:** $T \in \mathbb{R}^{d_1 \times \cdots \times d_N}$, for any $n \in [N]$, $T_{(n)} \in \mathbb{R}^{d_n \times d_1 \cdots d_{n-1} d_{n+1} \cdots d_N}$ ("mode-n matricization")

- **Remark:** $\|T\|_F = \|T_{(n)}\|_F$ for all $n \in [N]$

Def: If $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{k \times q}$, their **Kronecker product** $A \otimes B \in \mathbb{R}^{mk \times nq}$ is defined by:

$$A \otimes B = \begin{pmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{pmatrix} \quad (mk \times nq)$$

(a_{ij} is the entry (i,j) of A)

Remark: $a \otimes b = \underbrace{\underbrace{a}_{m\times 1} \otimes \underbrace{b}_{n\times 1}}_{mn \times 1} = \mathrm{vec}(ba^T) = \mathrm{vec}(\underbrace{b \circ a}_{m \times n})$

Def: The **Kathri-Rao product** of $A = \begin{pmatrix} | & & | \\ a_1 & \cdots & a_R \\ | & & | \end{pmatrix} \in \mathbb{R}^{m \times R}$ and $B = \begin{pmatrix} | & & | \\ b_1 & \cdots & b_R \\ | & & | \end{pmatrix} \in \mathbb{R}^{n \times R}$ is defined by

$$A \odot B = \begin{pmatrix} | & | & & | \\ a_1 \otimes b_1 & a_2 \otimes b_2 & \cdots & a_R \otimes b_R \\ | & | & & | \end{pmatrix} \in \mathbb{R}^{mn \times R}$$

Property: If $W = CP(A_1, A_2, \dots, A_N)$, then

$$W_{(n)} = A_n \underbrace{(A_N \odot \cdots \odot A_{n+1} \odot A_{n-1} \odot \cdots \odot A_1)^T}_{d_N \cdots d_{n+1} d_{n-1} \cdots d_1 \times R}$$

(A_n: $d_n \times R$; $W_{(n)}$: $d_n \times d_N \cdots d_{n+1} d_{n-1} \cdots d_1$)

Alternating least squares algorithm (ALS) for CP decomposition

$$\min_{A,B,C} \| T - CP(A,B,C) \|_F^2$$

A: $d_1 \times R$, B: $d_2 \times R$, C: $d_3 \times R$

We want solve this problem wrt A:

$$\underset{A}{\operatorname{argmin}} \| T - CP(A,B,C) \|_F^2 = \underset{A}{\operatorname{argmin}} \| T_{(1)} - \left(CP(A,B,C)\right)_{(1)} \|_F^2$$

(B, C fixed)

$$= \underset{A}{\operatorname{argmin}} \| T_{(1)} - A (C \odot B)^T \|_F^2$$

$$= T_{(1)} \left[(C \odot B)^T \right]^+$$

($^+$: pseudo-inverse)

(side note: if $A \in \mathbb{R}^{m \times n}$ with $m \leq n$ and $\operatorname{rank}(A) = m$, then $AA^+ = I$ (however: A^+A may not be I))

ALS ALGORITHM:

INPUT: target tensor T ($d_1 \times \cdots \times d_N$), rank R

OUTPUT: factor matrices A_i, $i = 1, \ldots, N$ s.t. $T \approx CP(A_1, \ldots, A_N)$

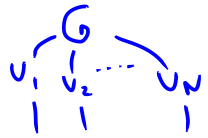
- Initialize $A_1, \ldots, A_N$
- Repeat
  - For $n = 1, \ldots, N$
    - $A_n \leftarrow T_{(n)} \left[\left(A_N \odot \cdots \odot A_{n+1} \odot A_{n-1} \odot \cdots \odot A_1 \right)^T \right]^+$
  - until convergence

(II) SVD-based algorithms for Tucker and TT

1) Tucker: Higher order SVD (Lauthauwer et al. 2000)

Given $T \in \mathbb{R}^{d_1 \times \cdots \times d_N}$ and $(R_1, \dots, R_N)$ we want to solve

$$\min_{\substack{G \in \mathbb{R}^{R_1 \times \cdots \times R_N} \\ U_i \in \mathbb{R}^{d_i \times R_i}, \\ i = 1 \dots N}} \| T - G \times_1 U_1 \times_2 U_2 \times \cdots \times_N U_N \|_F^2$$



HOSVD Algorithm

INPUT: $T \in \mathbb{R}^{d_1 \times \cdots \times d_N}$ and $(R_1, \dots, R_N)$

OUTPUT: $G \in \mathbb{R}^{R_1 \times \cdots \times R_N}$, $U_i \in \mathbb{R}^{d_i \times R_i}$, $i = 1 \dots N$ s.t. $T \approx G \times_1 U_1 \times_2 \cdots \times_N U_N$

For $n = 1 \dots N$

$U_n \leftarrow R_n$ leading left singular vectors of $T_{(n)}$ ($d_n \times R_n$)

↓ SVD

$U D V^T$

↓

extract the first R_n columns

$G \leftarrow T \times_1 U_1^T \times_2 \cdots \times_N U_N^T$ ($R_1 \times \cdots \times R_N$; U_1^T: $R_1 \times d_1$, U_N^T: $R_N \times d_N$; T: $d_1 \times \cdots \times d_N$)

RETURN $G, U_1, \dots, U_N$

$$\star \quad \min_X \| T - X \|_F^2 \quad \text{s.t.} \quad \text{rank}_{\text{TUCKER}}(X) \leq (R_1, \dots, R_N)$$

Theorem: Let X^* be the solution of problem $\star$ and X_{HOSVD} the solution returned by HOSVD. ($X_{\text{HOSVD}} = G \times_1 U_1 \times \cdots \times_N U_N$)

Then

$$\| T - X_{\text{HOSVD}} \|_F \leq \sqrt{N} \, \| T - X^* \|_F$$

- If T has Tucker rank less than $(R_1, \dots, R_N)$, then $\| T - X^* \|_F = 0$ hence $\| T - X_{\text{HOSVD}} \|_F = 0$ i.e. HOSVD recover the exact Tucker decomposition if it exists.

2) TENSOR TRAIN : TT-SVD (OSSEDELETS, 2010)

↳ (a.k.a. MATRIX PRODUCT STATE (MPS) : sequential SVD)

$$\min_{G_1, G_2, \dots G_N} \| T - TT(G_1, G_2, \dots, G_N) \|_F^2$$

↳ $G_1 \overset{R_1}{-} G_2 \overset{R_2}{-} G_3 \overset{R_3}{-} \cdots \overset{R_{N-1}}{-} G_N$ (dangling indices d_1, d_2, d_3, d_N)

TT- SVD Algorithm

INPUT : $T \in \mathbb{R}^{d_1 \times d_2 \times d_3 \times d_4}$, $(R_1, \dots, R_3)$

OUTPUT : $G_1, \dots, G_4$ s.t. $T \approx TT(G_1, G_2, G_3, G_4)$

- T (d_1 | $d_2 \times d_3 \times d_4$) $\approx$ G_1 (d_1, R_1) A ($R_1 d_2$ | $d_3 d_4$) (Low rank matrix approx. → SVD)
- A ($R_1 d_2$ | $d_3 d_4$) $\approx$ G_2 ($R_1 d_2$ | R_2) B ($R_2 d_3$ | d_4)
- B ($R_2 d_3$ | d_4) $\approx$ G_3 ($R_2 d_3$ | R_3) G_4 (R_3, d_4)
- RETURN G_1, G_2, G_3, G_4

Theorem : If X^* is the best approximation of T of TT rank $(R_1, \dots, R_{N-1})$ then $\| T - X_{TT\text{-}SVD} \|_F \leq \sqrt{N-1} \, \| T - X^* \|_F$.

↳ solution returned by TT-SVD

$X_{TT\text{-}SVD} = G_1 - G_2 - \cdots - G_N$

- If $\| T - X^* \|_F = 0$ then TT-SVD returns an exact TT decomposition of T.

(III) Supervised Learning With Tensor Networks

Supervised Learning With Quantum-Inspired Tensor Networks (NeurIPS, 2016)

E. Miles Stoudenmire[1,2] and David J. Schwab[3]

We want to learn a linear function $f : \mathbb{R}^{d^m} \longrightarrow \mathbb{R}^n$ (input space is very high dimension)

$\hookrightarrow f : x \longmapsto Wx$, with W of size $n \times d^m$

Feature map: Image $\longrightarrow \mathbb{R}^{d \times d \times \cdots \times d}$

$x \in \mathbb{R}^{h \times w} \longmapsto \phi(x_{11}) \otimes \phi(x_{12}) \otimes \phi(x_{13}) \otimes \cdots \otimes \phi(x_{hw})$

tensor of order hw $\hookrightarrow \mathbb{R}^{2^{hw}}$

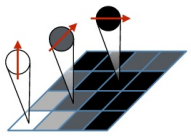
Two examples for ϕ are

- $\phi : \mathbb{R} \longrightarrow \mathbb{R}^2$, $\alpha \longmapsto \begin{pmatrix} \alpha \\ 1 \end{pmatrix}$

$\hookrightarrow (\alpha_1, \alpha_2, \alpha_3) \longmapsto \phi(\alpha_1) \otimes \phi(\alpha_2) \otimes \phi(\alpha_3) = \begin{pmatrix} \alpha_1 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} \alpha_2 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} \alpha_3 \\ 1 \end{pmatrix}$

Entries of this tensor: $1, \alpha_1, \alpha_2, \alpha_3, \alpha_1\alpha_2, \alpha_1\alpha_3, \alpha_2\alpha_3$ and $\alpha_1\alpha_2\alpha_3$

- $\phi : \mathbb{R} \longrightarrow \mathbb{R}^2$, $\alpha \longmapsto \begin{pmatrix} \cos\left(\frac{\pi}{2}\alpha\right) \\ \sin\left(\frac{\pi}{2}\alpha\right) \end{pmatrix}$



$f : \mathbb{R}^{d^m} \simeq \mathbb{R}^{\underbrace{d \times d \times \cdots \times d}_{m}} \longrightarrow \mathbb{R}^h$

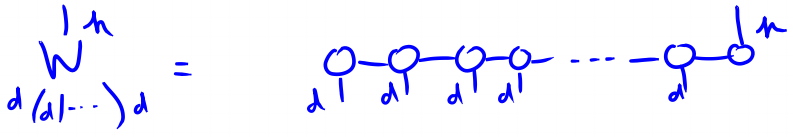
$X \in \mathbb{R}^{d \times d \times \cdots \times d}$ $\longmapsto$ $W \in \mathbb{R}^{d \times d \times \cdots \times d \times h}$

↳ The model parameter is $W \in \mathbb{R}^{d \times \cdots \times d \times h}$

↳ $d^m h$ parameters !

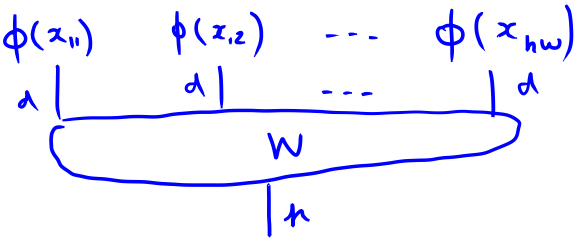
We cannot even store W in memory !

Sol: Parametrize W with a low rank TT decomposition

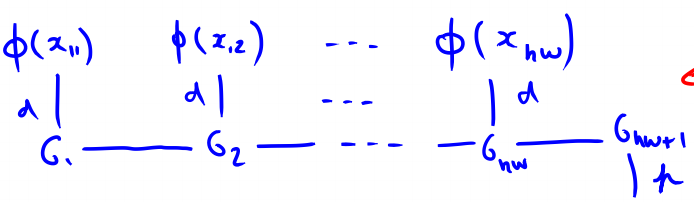


$f(X) = f(\phi(x_{11}) \circ \phi(x_{12}) \circ \cdots \circ \phi(x_{hw}))$

$= \phi(x_{11}) \otimes \phi(x_{12}) \otimes \cdots \otimes \phi(x_{hw}) \cdot W$



$= \phi(x_{11}) \otimes \phi(x_{12}) \otimes \cdots \otimes \phi(x_{hw}) \cdot G_1 G_2 \cdots G_{hw} G_{hw+1}$



← We can compute this very efficiently

Learning problem: Given data $\{(x_1, y_1), \cdots, (x_m, y_m)\} \subseteq \mathbb{R}^{d^m} \times \mathbb{R}^h$

we want to minimize $\mathcal{L} = \sum_{i=1}^{m} \ell(f(x_i), y_i)$ for some loss function $\ell : \mathbb{R}^h \to \mathbb{R}$.

↳ parameters: $G_1, G_2, \cdots, G_{hw+1}$ cores of a TT decomposition of W.

Training Algorithm (DMRG)

- Initialize $G_1, G_2, \dots, G_{m+1}$
- Repeat
  - for each consecutive pair of cores G_i, G_{i+1}
    - (merge two cores) $B = G_i \cdot G_{i+1}$
    - (Gradient descent step) $B^{new} = B - \gamma \nabla_B \mathcal{L}$
    - (split in two cores) $B^{new} \approx G_i^{new} \cdot G_{i+1}^{new}$ (truncated SVD)
    - $G_i, G_{i+1} \leftarrow G_i^{new}, G_{i+1}^{new}$
  - until convergence

* R_i^{new} can be chosen adaptively from the singular values of B^{new}.

Follow up papers:

From probabilistic graphical models to generalized tensor networks for supervised learning

Ivan Glasser,[1,2] Nicola Pancotti,[1,2] and J. Ignacio Cirac[1,2]

Tensor Networks for Probabilistic Sequence Modeling

Jacob Miller
Mila and DIRO
Université de Montréal
Montréal QC, Canada
jmjacobmiller@gmail.com

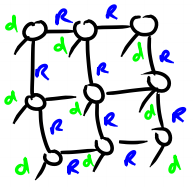
Guillaume Rabusseau
Mila and DIRO
Université de Montréal
Montréal QC, Canada
grabus@iro.umontreal.ca

John Terilla
CUNY and Tunnel
City University of New York
New York NY, USA
jterilla@gc.cuny.edu

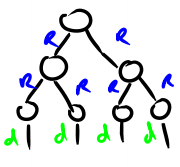
IV BEYOND TUCKER, TT, ...

TT:

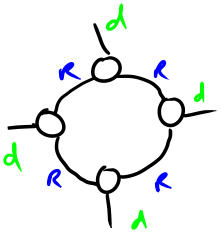
PEPS (2d in TN):



HIERARCHICAL TUCKER: (Tree TN)



TENSOR RING:



TENSOR METHODS FOR ML

→ COMPRESS NEURAL NETWORKS

→ LEARNING LATENT VARIABLE MODELS

(I) EFFICIENT COMPUTATIONS WITH TENSOR NETWORKS

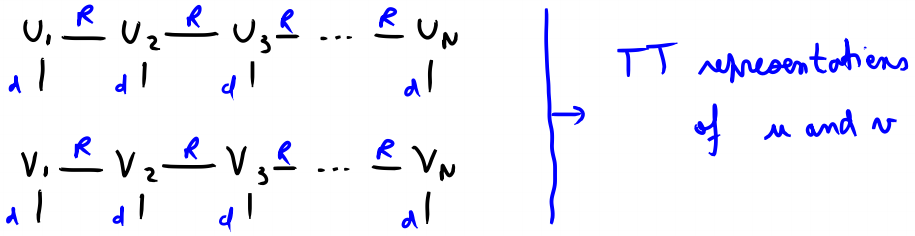
- **Inner product**: $u, v \in \mathbb{R}^{d^N}$

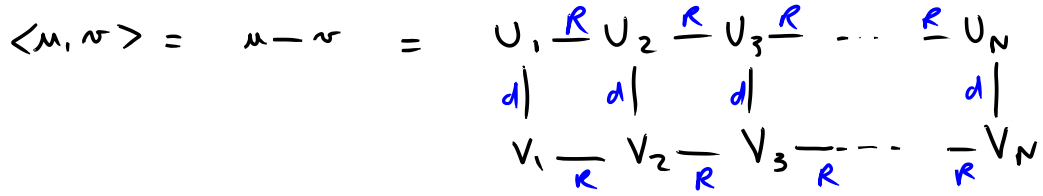
  What is the computational cost for $\langle u, v \rangle = \sum_{i=1}^{d^N} u_i v_i$?

  ↳ $\mathcal{O}(d^N)$

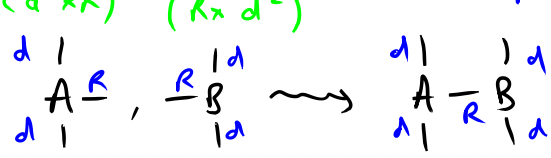
Suppose u and v are in the TT format:

$$u, v \in \mathbb{R}^{d^N} \cong \mathbb{R}^{\underbrace{d \times d \times \cdots \times d}_{N \text{ times}}}$$



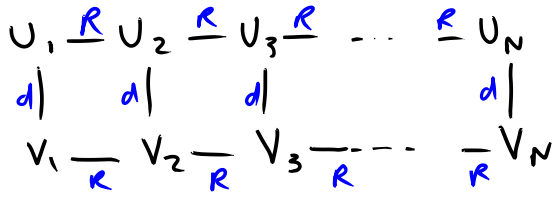


Recall: Computing the matrix product AB has cost $\mathcal{O}(mnp)$ ($m \times n$, $n \times p$)

For ex, Computing  can be done

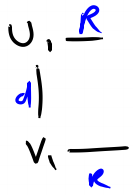
with a matrix product between matrices of shapes $d^2 \times R$ and $R \times d^2$. Hence the complexity is $\mathcal{O}(Rd^4)$.

We want to compute



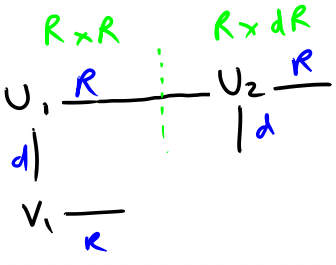
COMPLEXITY

① Compute



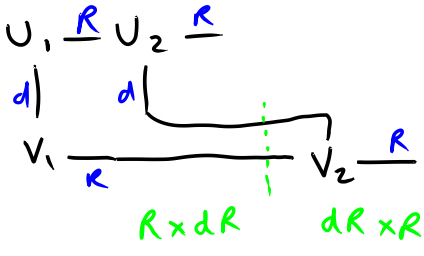
$\rightsquigarrow R \times d \quad d \times R$ $\quad O(dR^2)$

②



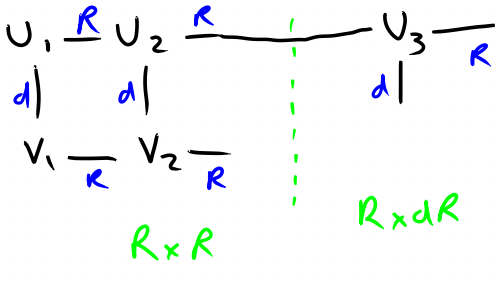
$O(dR^3)$

③



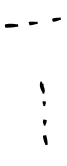
$O(dR^3)$

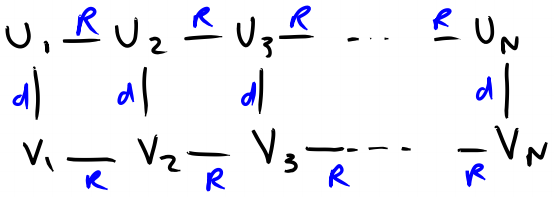
④



$O(dR^3)$

⑤





this is linear in N

TOTAL COMPLEXITY $O(NdR^3)$

↳ in contrast with $O(d^N)$!

exp. in N

- **Matrix vector product**

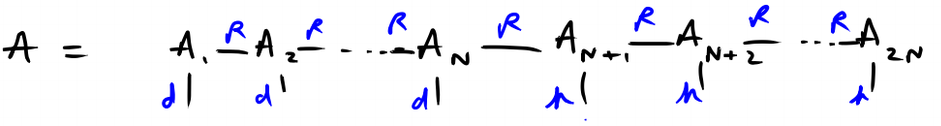
$A \in \mathbb{R}^{d^N \times n^N}$, $x \in \mathbb{R}^{n^N}$: we want to compute $b = Ax \in \mathbb{R}^{d^N}$.

In fact, we want to compute b in the TT format.

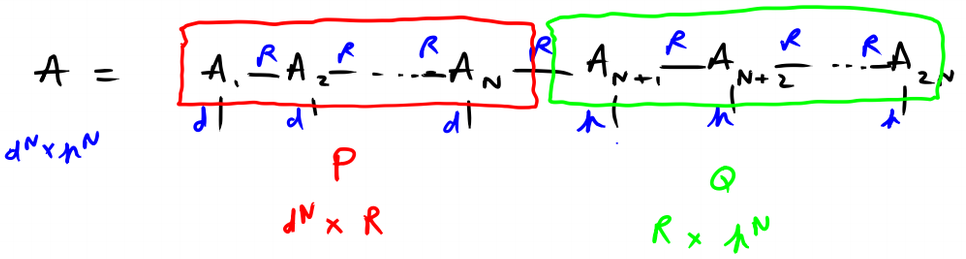
- TT representation of matrices :

$$A \in \mathbb{R}^{d^N \times n^N} \cong \mathbb{R}^{\underbrace{d \times d \times \dots \times d}_{N \text{ times}} \times \underbrace{n \times n \times \dots \times n}_{N \text{ times}}}$$

① First solution :

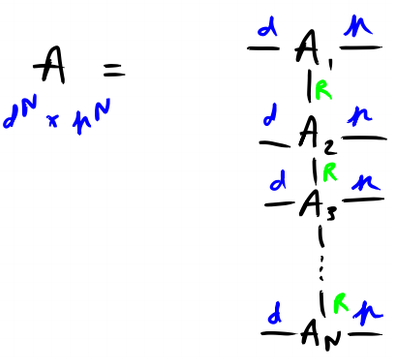


↳ PBM : A will be of rank at most R



↳ $A = PQ$ → Rank R factorization of A, hence $\text{rank}(A) \leq R$.

② TT matrix representation :



↳ Remark • If $A = \underset{d^2 \times n^2}{\;}$ $\begin{matrix} A_1 \\ A_2 \end{matrix}$ and $R = 1$,

then $A = \underset{d^2 \times n^2}{\;}$ $\begin{matrix} A_1 \\ A_2 \end{matrix}$ $= A_1 \otimes A_2$ (⊗ : KRONECKER PRODUCT)

$d \times n \quad d \times n$

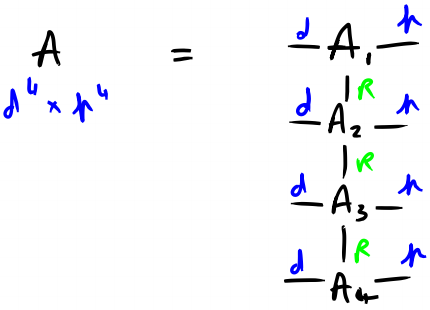
• In a TN, an edge of dimension 1 is equivalent to having no edge :

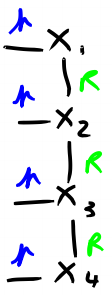
$$\left(\underset{m}{B} \; \underset{1}{C} \right)_{ij} = \sum_{k=1}^{1} B_{ik} C_{kj} = B_{i1} C_{1j} = b_i c_j$$

B : $m \times 1$ vector b, C : $1 \times m$ vector c^T

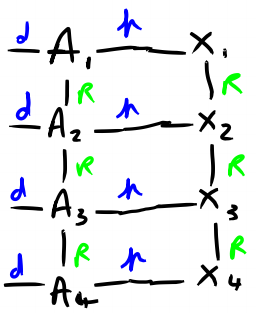
$= b \, c^T$

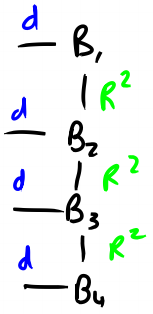
Back to matrix vector product:

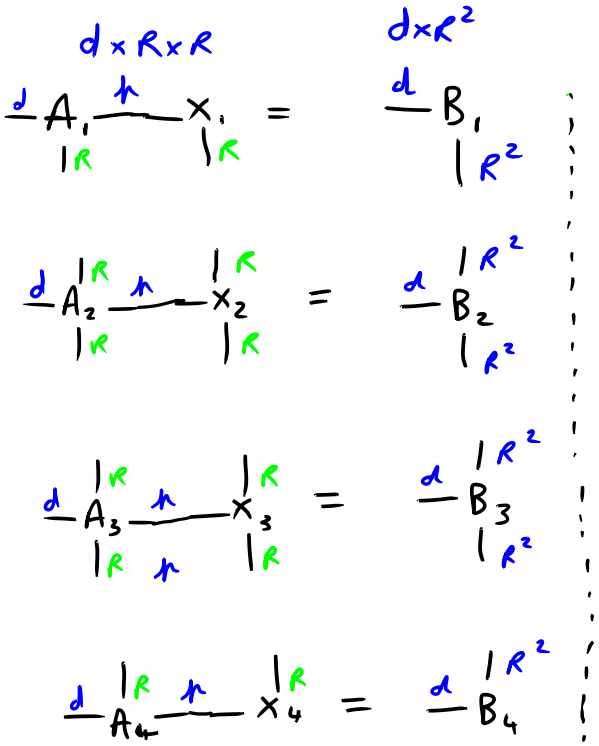
$A \in \mathbb{R}^{d^4 \times n^4}$ = 

$x \in \mathbb{R}^{n^4}$ = 

We want a TT representation of $b = Ax$

$b \in \mathbb{R}^{d^4}$ = 

b = 



$\Rightarrow$ If $A \in \mathbb{R}^{d^N \times n^N}$ given as a TT matrix with rank R

and $x \in \mathbb{R}^{n^N}$ given as a TT vector with rank R

then a rank R^2 TT representation of $b = Ax$ can be computed in time $\mathcal{O}(NR^4 d n)$.

↳ in contrast with $\mathcal{O}(d^N n^N)$

- $u, v \in \mathbb{R}^{d^N}$, given as rank R TT-vector, a rank $2R$ TT representation of $u + v$ can be computed in $\mathcal{O}(NR^3 d)$.

- $u, v \in \mathbb{R}^{d^N}$, given as rank R TT-vector, a rank R^2 TT representation of $u \circledast v$ can be computed in $\mathcal{O}(NR^3 d)$.
  ↳ component-wise product.

- TT ROUNDING:

  Given a rank R representation of $u \in \mathbb{R}^{d^N}$, the TT rounding operation returns a rank $\hat{R}$ TT representation of u which is "close" to the original representation (in time $\mathcal{O}(NR^3 d)$).

FOR MORE DETAILS:

SIAM J. SCI. COMPUT.
Vol. 33, No. 5, pp. 2295–2317

TENSOR-TRAIN DECOMPOSITION*

I. V. OSELEDETS†

Abstract. A simple nonrecursive form of the tensor decomposition in d dimensions is presented. It does not inherently suffer from the curse of dimensionality, it has asymptotically the same number of parameters as the canonical decomposition, but it is stable and its computation is based on low-rank approximation of auxiliary *unfolding matrices*. The new form gives a clear and convenient way to implement all basic operations efficiently. A fast rounding procedure is presented, as well as basic linear algebra operations. Examples showing the benefits of the decomposition are given, and the efficiency is demonstrated by the computation of the smallest eigenvalue of a 19-dimensional operator.

(11) COMPRESSING NEURAL NETWORKS

Tensorizing Neural Networks

(NeurIPS, 2015)

Alexander Novikov[1,4] **Dmitry Podoprikhin**[1] **Anton Osokin**[2] **Dmitry Vetrov**[1,3]
[1]Skolkovo Institute of Science and Technology, Moscow, Russia
[2]INRIA, SIERRA project-team, Paris, France
[3]National Research University Higher School of Economics, Moscow, Russia
[4]Institute of Numerical Mathematics of the Russian Academy of Sciences, Moscow, Russia
novikov@bayesgroup.ru podoprikhin.dmitry@gmail.com
anton.osokin@inria.fr vetrovd@yandex.ru

Fully connected layer: $y = \sigma(Wx + b)$, $y \in \mathbb{R}^M$, $W \in \mathbb{R}^{M \times N}$, $b \in \mathbb{R}^M$, $x \in \mathbb{R}^N$, σ = activation function

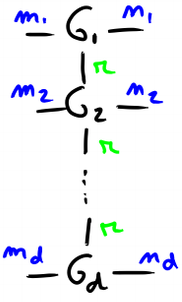
size: $M \times N$

For sake of simplicity, we ignore the bias term: $y = \sigma(Wx)$

IDEA: Represent W as a TT matrix.

$M = m_1 \times m_2 \times \cdots \times m_d$

$N = n_1 \times n_2 \times \cdots \times n_d$ (d is the order of the tensor)

W ($M \times N$) = 

TT layer

↳ parameters are $G_1, G_2, \ldots, G_d$

COMPLEXITY:

Operation	Time	Memory
FC forward pass	$O(MN)$	$O(MN)$
TT forward pass	$O(dr^2 m \max\{M, N\})$	$O(r \max\{M, N\})$
FC backward pass	$O(MN)$	$O(MN)$
TT backward pass	$O(d^2 r^4 m \max\{M, N\})$	$O(r^3 \max\{M, N\})$

Table 1: Comparison of the asymptotic complexity and memory usage of an $M \times N$ TT-layer and an $M \times N$ fully-connected layer (FC). The input and output tensor shapes are $m_1 \times \ldots \times m_d$ and $n_1 \times \ldots \times n_d$ respectively ($m = \max_{k=1\ldots d} m_k$) and r is the maximal TT-rank.

EXPERIMENT ON CIFAR 10:

INPUT (3072) → TT-layer ($3072 \to 2^{18}$, $\sim 250{,}000$) → TT-layer ($2^{18} \to 2^{12}$, 4096) → FC layer ($2^{12} \to 10$) → OUTPUT (10)

- COMPUTING GRADIENTS OF TENSOR NETWORKS

↳ See TN-gradients.pdf

Tensor Decomposition for Learning Latent Variable Models (LVM)

Journal of Machine Learning Research 15 (2014) 2773-2832 Submitted 2/13; Revised 3/14; Published 8/14

Tensor Decompositions for Learning Latent Variable Models

Animashree Anandkumar A.ANANDKUMAR@UCI.EDU
Electrical Engineering and Computer Science
University of California, Irvine
2200 Engineering Hall
Irvine, CA 92697

Rong Ge RONGGE@MICROSOFT.COM
Microsoft Research
One Memorial Drive
Cambridge, MA 02142

Daniel Hsu DJHSU@CS.COLUMBIA.EDU
Department of Computer Science
Columbia University
1214 Amsterdam Avenue, #0401
New York, NY 10027

Sham M. Kakade SKAKADE@MICROSOFT.COM
Microsoft Research
One Memorial Drive
Cambridge, MA 02142

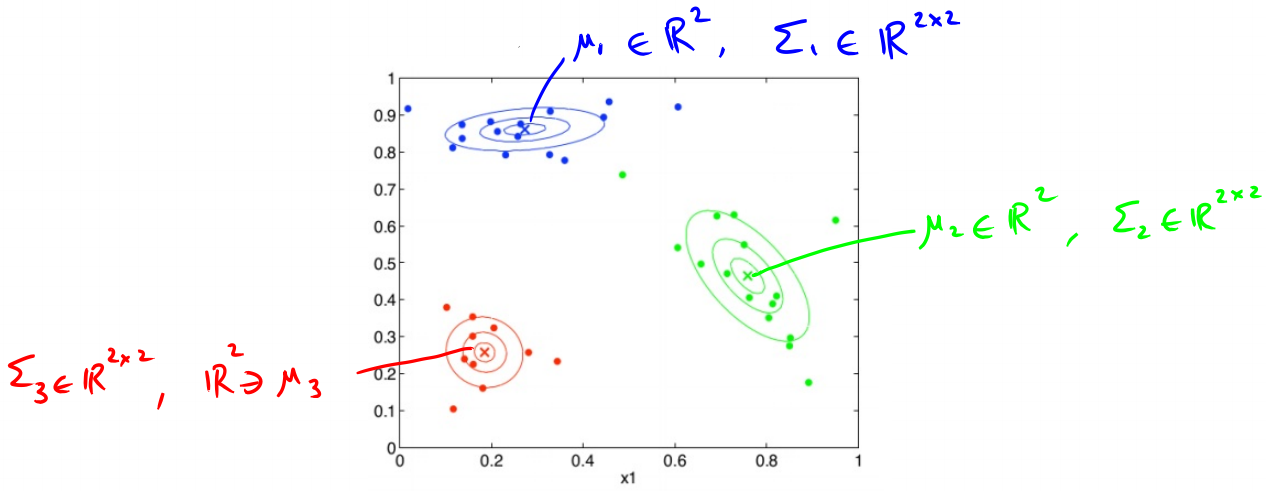
Matus Telgarsky MTELGARS@CS.UCSD.EDU
Department of Statistics
Rutgers University
110 Frelinghuysen Road
Piscataway, NJ 08854

Editor: Benjamin Recht

(JMLR, 2014)

1) Examples of LVM

- Gaussian mixture model (GMM)



GMM with k components in $\mathbb{R}^d$:

- parameters: $\mu_1, \dots, \mu_k \in \mathbb{R}^d$ (centers)
  $\Sigma_1, \dots, \Sigma_k \in \mathbb{R}^{d\times d}$ ← p.s.d. matrices (covariance matrices)
  $h_1, \dots, h_k \in \mathbb{R}_+$, $0 \le h_i \le 1$, $\sum_{i=1}^{k} h_i = 1$ (mixing probabilities)
- data generating distribution

Draw $x \sim$ GMM ←
(i) Draw a Gaussian h with probabilities $\mathbb{P}(h = i) = h_i$, $i = 1, \dots, k$
(ii) Draw $x \sim \mathcal{N}(\mu_i, \Sigma_i)$

- **Single topic model**

(Latent variable is the topic of a document)

Topic: discrete random variable taking values $1, 2, \dots, k$

Vocabulary of size d (d different words).

→ We want to model the distribution over the vocabulary given a topic.

- Parameters:

$\mathbb{P}(\text{topic} = i) = \pi_i$ $\quad \pi_1, \pi_2, \dots, \pi_k \in \mathbb{R}$ (topic probabilities) $\quad 0 \le \pi_i \le 1$, $\sum_{i=1}^{k} \pi_i = 1$

$\mathbb{P}(\text{word} = j \mid \text{topic} = i) = (\mu_i)_j$ $\quad \mu_1, \mu_2, \dots, \mu_k \in \mathbb{R}^d$ (voc. distrib. given a topic)

for $i = 1, \dots, k$: $\quad 0 \le (\mu_i)_j \le 1$, $\sum_{j=1}^{n} (\mu_i)_j = 1$

- Data generating distribution

Draw a document of length ℓ from the single topic model.

- Draw a topic h with probabilities $\mathbb{P}(h = i) = \pi_i$
- Draw independently ℓ words with probabilities $\mathbb{P}(\text{word } j \mid \text{topic} = h) = (\mu_i)_j$

2) Moments and Latent variable models

Def: • If x is a random variable taking its values in $\mathbb{R}$, its n^{th} order moment is $\mathbb{E}[x^n]$.

• If x is a random variable taking its values in $\mathbb{R}^d$, its n^{th} order moment is $\mathbb{E}[x^{\otimes n}] = \mathbb{E}[\underbrace{x \otimes x \otimes \cdots \otimes x}_{n \text{ times}}] \in \mathbb{R}^{\underbrace{d \times d \times \cdots \times d}_{n \text{ times}}}$

↳ . 3rd moment : $\mathbb{E}[x \otimes x \otimes x]_{i,j,k} = \mathbb{E}[x_i x_j x_k]$

. 1st moment : $\mathbb{E}[x] \rightarrow$ mean

. 2nd moment : $\mathbb{E}[x \otimes x] = \mathbb{E}[x x^T]$

↳ if $\mathbb{E}[x] = 0$, this is the covariance matrix.

↳ $\mathbb{V}[x] = \mathbb{E}[x^2] - \mathbb{E}[x]^2$

$Cov[x] = \mathbb{E}[xx^T] - \mathbb{E}[x]\mathbb{E}[x]^T$

→ the method of moments give the same estimators as the maximum likelihood for a Gaussian distribution.

If $x \sim \mathcal{N}(\mu, \sigma^2)$ $\quad \mathbb{E}[x] = \mu$, $\mathbb{E}[x^2] = \sigma^2 + \mu^2$ $\quad \rightarrow \quad \mu = \mathbb{E}[x] \approx \hat{\mu} = \frac{1}{m}\sum_{i=1}^{m} x_i$, $\sigma^2 = \mu^2 - \mathbb{E}[x^2] \approx \hat{\mu}^2 - \frac{1}{m}\sum_{i=1}^{m} x_i^2$

Single Topic Model

parameters: $\theta = \{\pi_1, \pi_2, \ldots, \pi_k, \mu_1, \ldots, \mu_k\}$ ($\pi_h \in \mathbb{R}$, $\mu_h \in \mathbb{R}^d$)

$$\mathbb{P}(\text{word } i,j \mid \text{topic } h) = \mathbb{P}(\text{word } i \mid \text{topic } h)\,\mathbb{P}(\text{word } j \mid \text{topic } h)$$
$$= (\mu_h)_i\,(\mu_h)_j$$
$$= (\mu_h \otimes \mu_h)_{i,j}$$

$$\mathbb{P}(\text{word } i,j) = \sum_{h=1}^{k} \mathbb{P}(\text{topic } h)\,\mathbb{P}(\text{word } i,j \mid \text{topic } h)$$
$$= \sum_{h=1}^{k} \pi_h\,(\mu_h \otimes \mu_h)_{i,j}$$

$$\mathbb{P}(\text{word } i_1, i_2, i_3) = \sum_{h=1}^{k} \pi_h\,(\mu_h \otimes \mu_h \otimes \mu_h)_{i_1,i_2,i_3}$$

Note that if x_1, x_2 and x_3 are the one-encodings of the first 3 words in a document, then

$$\mathbb{E}[x_1 \otimes x_2]_{i_1,i_2} = \mathbb{P}(\text{word } i_1, i_2)$$
$$\mathbb{E}[x_1 \otimes x_2 \otimes x_3]_{i_1,i_2,i_3} = \mathbb{P}(\text{word } i_1, i_2, i_3)$$

Theorem 3.1 (Anandkumar et al., 2012c) *If*

$$\begin{aligned} M_2 &:= \mathbb{E}[x_1 \otimes x_2] \\ M_3 &:= \mathbb{E}[x_1 \otimes x_2 \otimes x_3], \end{aligned}$$

then

$$\begin{aligned} M_2 &= \sum_{i=1}^{k} w_i \, \mu_i \otimes \mu_i \\ M_3 &= \sum_{i=1}^{k} w_i \, \mu_i \otimes \mu_i \otimes \mu_i. \end{aligned}$$

$(w_i = h_i)$

GMM

parameters : $\theta = \{ \mu_1, \dots, \mu_k, \Sigma_1, \dots, \Sigma_k, w_1, \dots, w_k \}$

$\mu_1, \dots, \mu_k$: centers $\in \mathbb{R}^d$

$\Sigma_1, \dots, \Sigma_k$: covariance matrices $\in \mathbb{R}^{d \times d}$

$w_1, \dots, w_k$: mixing probabilities

↳ Here we assume each $\Sigma_i = \sigma_i^2 I$ for each $i = 1, \dots, k$

↳ $\sigma_i \in \mathbb{R}$

Theorem 3.2 (Hsu and Kakade, 2013) *Assume $d \geq k$. The variance σ^2 is the smallest eigenvalue of the covariance matrix $\mathbb{E}[x \otimes x] - \mathbb{E}[x] \otimes \mathbb{E}[x]$. Furthermore, if*

$$\begin{aligned} M_2 &:= \mathbb{E}[x \otimes x] - \sigma^2 I \\ M_3 &:= \mathbb{E}[x \otimes x \otimes x] - \sigma^2 \sum_{i=1}^{d} (\mathbb{E}[x] \otimes e_i \otimes e_i + e_i \otimes \mathbb{E}[x] \otimes e_i + e_i \otimes e_i \otimes \mathbb{E}[x]), \end{aligned}$$

then

$$\begin{aligned} M_2 &= \sum_{i=1}^{k} w_i \, \mu_i \otimes \mu_i \\ M_3 &= \sum_{i=1}^{k} w_i \, \mu_i \otimes \mu_i \otimes \mu_i. \end{aligned}$$

$\sigma = \sigma_1 = \sigma_2 = \dots = \sigma_k$

(e_i : i-th vector of the canonical basis)

Summary : In both cases, there exists $M_2 \in \mathbb{R}^{d \times d}$ and a tensor $M_3 \in \mathbb{R}^{d \times d \times d}$ (which can be estimated from data) satisfying

$$\begin{aligned} M_2 &= \sum_{i=1}^{k} w_i \, \mu_i \otimes \mu_i \\ M_3 &= \sum_{i=1}^{k} w_i \, \mu_i \otimes \mu_i \otimes \mu_i. \end{aligned}$$

↳ CP decomposition of M_3 (under mild conditions, this decomposition unique)

Decomposition of M_2 and M_3:

$$M_2 = \sum_{i=1}^{k} w_i \, \mu_i \otimes \mu_i$$

$$M_3 = \sum_{i=1}^{k} w_i \, \mu_i \otimes \mu_i \otimes \mu_i.$$

Observation : • The problem of orthogonal tensor decomposition,

$$T = \sum_{i=1}^{k} w_i \, \mu_i \otimes \mu_i \otimes \mu_i \quad \text{where} \quad \langle \mu_i, \mu_j \rangle = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{o.w.} \end{cases}$$

is easy to solve.

• We can use M_2 to "orthogonalize" M_3.

$M_3 \longrightarrow \tilde{M}_3 \longrightarrow \tilde{w}_i, \tilde{\mu}_i \longrightarrow w_i, \mu_i$

$M_3 \longrightarrow \tilde{M}_3$: using M_2

$\tilde{M}_3 \longrightarrow \tilde{w}_i, \tilde{\mu}_i$: orthogonal tensor decomposition

$$\tilde{M}_3 = \sum_{i=1}^{k} \tilde{w}_i \, \tilde{\mu}_i \otimes \tilde{\mu}_i \otimes \tilde{\mu}_i$$

COMPUTING GRADIENTS OF TENSOR NETWORKS

(I) Jacobian and backpropagation

- $f: \mathbb{R}^m \to \mathbb{R}$ **Gradient** $\nabla_\theta f = \begin{pmatrix} \partial f/\partial \theta_1 \\ \vdots \\ \partial f/\partial \theta_m \end{pmatrix}$ for each $\theta \in \mathbb{R}^m$

- $f: \mathbb{R}^m \to \mathbb{R}^n$ **Jacobian** $\frac{\partial f}{\partial \theta} = \left(\frac{\partial f(\theta)_i}{\partial \theta_j} \right)_{i,j} \in \mathbb{R}^{n \times m}$ for each $\theta \in \mathbb{R}^m$

- **Backpropagation** / Automatic Differentiation (reverse mode).

Computational Graph $\quad \theta \to T^1 \to T^2 \to T^3 \to \mathcal{L}$ (where $X \to Y$ means that Y is a function of X)

$\mathbb{R}^m \to \mathbb{R}^{d_1} \to \mathbb{R}^{d_2} \to \mathbb{R}^{d_3} \to \mathbb{R}$

We want to compute $\frac{\partial \mathcal{L}}{\partial \theta}$.

Chain rule: $\frac{\partial \mathcal{L}}{\partial \theta} = \frac{\partial \mathcal{L}}{\partial T_3} \frac{\partial T_3}{\partial T_2} \frac{\partial T_2}{\partial T_1} \frac{\partial T_1}{\partial \theta}$

$\bar{\theta}$: $1 \times m$ $\quad 1 \times d_3$ $\quad d_3 \times d_2$ $\quad d_2 \times d_1$ $\quad d_1 \times m$

We define the adjoint: $\bar{T} = \frac{\partial \mathcal{L}}{\partial T}$

$\bar{T}_3 = \frac{\partial \mathcal{L}}{\partial T_3}$

$\bar{T}_2 = \frac{\partial \mathcal{L}}{\partial T_2} = \frac{\partial \mathcal{L}}{\partial T_3} \frac{\partial T_3}{\partial T_2} = \bar{T}_3 \frac{\partial T_3}{\partial T_2}$

$\bar{T}_1 = \frac{\partial \mathcal{L}}{\partial T_1} = \frac{\partial \mathcal{L}}{T_2} \frac{\partial T_2}{\partial T_1} = \bar{T}_2 \frac{\partial T_2}{\partial T_1}$

$\bar{\theta} = \cdots = \bar{T}_1 \frac{\partial T_1}{\partial \theta}$

$\longmapsto$ $\boxed{\bar{T}^j = \bar{T}^{j+1} \frac{\partial T^{j+1}}{\partial T^j}}$

More complex graphs: $\theta \to T_1 \to T_2, T_3 \to T_4 \to \mathcal{L}$

$$\overline{T_i} = \sum_{j : \text{child of } i} \overline{T_j} \frac{\partial T_j}{\partial T_i}$$

ex: $\overline{T_1} = \frac{\partial \mathcal{L}}{\partial T_1} = \frac{\partial \mathcal{L}}{\partial T_2}\frac{\partial T_2}{\partial T_1} + \frac{\partial \mathcal{L}}{\partial T_3}\frac{\partial T_3}{\partial T_1}$

$= \overline{T_2}\frac{\partial T_2}{\partial T_1} + \overline{T_3}\frac{\partial T_3}{\partial T_1}$

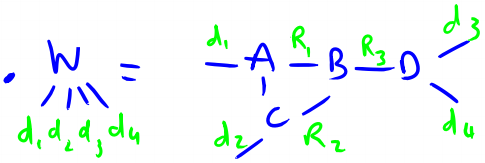
- $f : \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_N} \to \mathbb{R}^{h_1 \times h_2 \times \cdots \times h_P}$ $\left(\cong \mathbb{R}^{n_1 n_2 \cdots n_N} \to \mathbb{R}^{h_1 h_2 \cdots h_P}\right)$

Jacobian Tensor $\quad \frac{\partial f(T)}{\partial T} = \left(\frac{\partial f(T)_{i_1,\dots i_P}}{\partial T_{j_1,\dots j_N}}\right)_{i_1,\dots i_P j_1,\dots j_N} \in \mathbb{R}^{h_1 \times \cdots \times h_P \times n_1 \times \cdots \times n_N}$

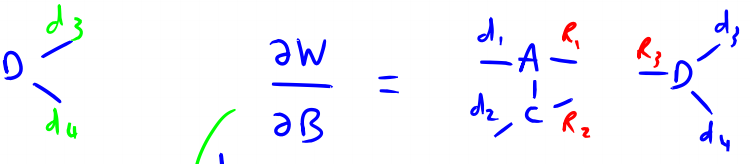
II Jacobian of Tensor Networks

Let W be a tensor given as a tensor network where G is a core tensor appearing only once. Then $\frac{\partial W}{\partial G}$ is simply obtained by deleting G in the tensor network.

Ex: • W =



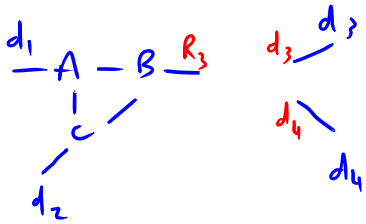
$\frac{\partial W}{\partial B}$ =



$d_1 \times d_2 \times d_3 \times d_4 \times R_1 \times R_2 \times R_3$

→ Jacobian of $B \mapsto W$ $\quad R_1 \times R_2 \times R_3 \times d_1 \times d_2 \times d_3 \times d_4$

$\frac{\partial W}{\partial D}$ =



$d_1 \times d_2 \times d_3 \times d_4 \times R_3 \times d_3 \times d_4$

• Rederiving classical matrix / vector gradients:

$$\frac{\partial \langle u, v \rangle}{\partial u} = \partial(u - v)/\partial u = -v = v$$

$$\frac{\partial Ax}{\partial x} = \partial(-A-x)/\partial x = -A- \quad = A$$

$$\frac{\partial x^T A x}{\partial A} = \partial(x-A-x)/\partial A = x- \quad -x \quad = xx^T$$

$$\frac{\partial \mathrm{Tr}(A)}{\partial A} = \partial(\textcircled{A})/\partial A = \quad \cap \quad = I$$

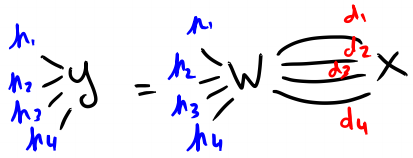
$$\frac{\partial Ax}{\partial A} = \partial(-A-x)/\partial A = \quad - \quad -x \cong I \circ x$$

If the core tensor G appears k times in the tensor network of W, then $\frac{\partial W}{\partial G}$ is given the sum of k copies of W where a different occurence of G is deleted in each copy.

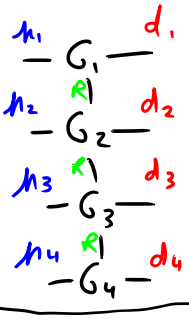
Ex: $\frac{\partial\, x^T A x}{\partial x} = \partial(x_1 A_2 x)/\partial x = x_1 A_2 + x_1 A_2 = (A + A^T)x$

Applications

TENSORIZING NEURAL NETWORKS :



where

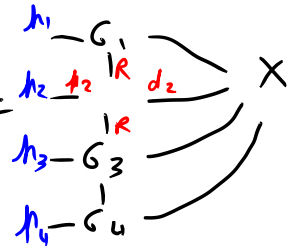


↳ We want to compute $\frac{\partial \ell}{\partial G^2} = \frac{\partial \ell}{\partial y}\frac{\partial y}{\partial G_2}$ ← Some loss depending on W

Comp. Graph: 

We need to compute $\frac{\partial y}{\partial G_2} \in \mathbb{R}^{h_1 \times h_2 \times h_3 \times h_4 \times R \times h_2 \times d_2 \times R}$. We have :

$$\frac{\partial y}{\partial G_2} = \partial\left(\begin{array}{c} G_1 \\ G_2 \\ G_3 \\ G_4 \end{array} \right) X \Big) / \partial G_2 =$$



IV Generalized CP decomposition

GENERALIZED CANONICAL POLYADIC TENSOR DECOMPOSITION*

DAVID HONG†, TAMARA G. KOLDA‡, AND JED A. DUERSCH§

↳ Likelihood maximization & generalized linear model.

Observations/Data : $X \in \mathbb{R}^{m_1 \times \cdots \times m_d}$

Parameterized PDF : $X_{i_1 \ldots i_d} \sim p(X_{i_1 \cdots i_d} \mid \theta_{i_1 \cdots i_d})$

where $\ell(\theta_{i_1 \cdots i_d}) = M_{i_1 \cdots i_d}$

$\ell : \mathbb{R} \to \mathbb{R}$ is an invertible link function connecting the model parameters $M_{i_1 \cdots i_d}$ to the "natural" parameter of the distribution.

Model parameters : $M \in \mathbb{R}^{m_1 \times \cdots \times m_d}$

Goal : Find M that maximizes the likelihood of the observed entries of X.

↳ $\Omega \subseteq [m_1] \times \cdots \times [m_d]$

$$\max_M \mathcal{L}(M; X, \Omega) \equiv \prod_{(i_1 \cdots i_d) \in \Omega} p(X_{i_1 \cdots i_d} \mid \theta_{i_1 \cdots i_d})$$

$$\text{where } \ell(\theta_{i_1 \cdots i_d}) = M_{i_1 \cdots i_d}$$

→ We only have (at most) one observation for each entry of X!

We will assume a low rank structure of M (interdependence between the $\theta_{i_1 \cdots i_d}$'s)

↳ low CP rank of M.

Reformulation of optimization problem :

$$\min_M F(M; X, \Omega) = \sum_{(i_1 \cdots i_d) \in \Omega} f(X_{i_1 \cdots i_d}, M_{i_1 \cdots i_d}) \quad \text{s.t. } \operatorname{rank}_{CP}(M) \leq R$$

$$\text{where } f(x, m) = -\log p(x \mid \ell^{-1}(m))$$

↑ "loss function" (assumed differentiable)

Examples of data distribution

① Gaussian distribution (Gaussian MLE $\Leftrightarrow$ Mean squared error)

$$X_{i,\dots,id} = M_{i,\dots,id} + \varepsilon_{i,\dots,id} \quad \text{where } \varepsilon_{i,\dots,id} \sim \mathcal{N}(0, \sigma^2)$$

(σ^2: constant)

Equivalently: $X_{i,\dots,id} \sim \mathcal{N}(\theta_{i,\dots,id}, \sigma^2)$ where $\theta_{i,\dots,id} = M_{i,\dots,id}$

the link function is the identity! $f(\theta) = \theta$

Simple derivation:

$$f(x, m) = -\log \mathcal{N}(x; m, \sigma^2) = \frac{(x-m)^2}{2\sigma^2} + \frac{1}{2}\log(2\pi\sigma^2)$$

↳ ignoring σ^2 and constants $\boxed{f(x, m) = (x-m)^2}$

$\Rightarrow$ ★ is then equivalent to CP decomposition.

② Bernoulli distribution.

If $X_{i,\dots,id} \in \{0,1\}$ (binary data), we can assume $X_{i,\dots,id} \sim \mathcal{B}(\theta_{i,\dots,id})$

$$h(x|\theta) = \theta^x (1-\theta)^{1-x} \quad , x \in \{0,1\}, \ \theta \in [0,1]$$

link function?

- identity $\rightsquigarrow$ we need to constrain $m \in [0,1]$
- odds ratio: $m = f(\theta) = \frac{\theta}{1-\theta}$

  $\rightsquigarrow$ only need to enforce $m \geq 0$
- log odds ratio:

  $m = f(\theta) = \log\left(\frac{\theta}{1-\theta}\right)$

  ↳ $\theta = \frac{e^m}{1+e^m}$, $1-\theta = \frac{1}{1+e^m}$

$$\dots \quad f(x, m) = -\log h(x|\theta) = \underbrace{-x\log\theta - (1-x)\log(1-\theta)}_{\text{cross-entropy}}$$

$$\boxed{f(x, m) = \log(1+e^m) - xm}$$

③ Other distributions

Table 1: Statistically-motivated loss functions. Parameters in blue are assumed to be constant. Numerical adjustments are indicated in red.

Distribution	Link function	Loss function	Constraints
$\mathcal{N}(\mu,\sigma)$	$m=\mu$	$(x-m)^2$	$x,m\in\mathbb{R}$
Gamma(k,σ)	$m=k\sigma$	$x/(m+\epsilon)+\log(m+\epsilon)$	$x>0, m\geq 0$
Rayleigh(θ)	$m=\sqrt{\pi/2}\,\theta$	$2\log(m+\epsilon)+(\pi/4)(x/(m+\epsilon))^2$	$x>0, m\geq 0$
Poisson(λ)	$m=\lambda$	$m-x\log(m+\epsilon)$	$x\in\mathbb{N}, m\geq 0$
	$m=\log\lambda$	e^m-xm	$x\in\mathbb{N}$, $m\in\mathbb{R}$
Bernoulli(ρ)	$m=\rho/(1-\rho)$	$\log(m+1)-x\log(m+\epsilon)$	$x\in\{0,1\}, m\geq 0$
	$m=\log(\rho/(1-\rho))$	$\log(1+e^m)-xm$	$x\in\{0,1\}$, $m\in\mathbb{R}$
NegBinom(r,ρ)	$m=\rho/(1-\rho)$	$(r+x)\log(1+m)-x\log(m+\epsilon)$	$x\in\mathbb{N}$, $m\geq 0$

Gradient based optimization

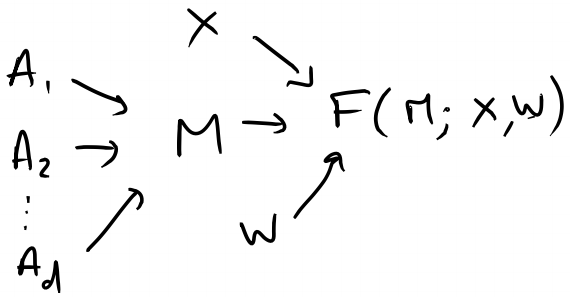
(if Ω is the set of observed entries: $W_{i_1,\dots,i_d}=1$ if $(i_1,\dots,i_d)\in\Omega$ and 0 o.w.)

$m_1\times\cdots\times m_d$

$$\star: \quad \min_{A_1,\dots,A_d} F(\mathcal{M};\mathcal{X},\mathcal{W}) \equiv \sum_{i_1,\dots,i_d} W_{i_1,\dots,i_d}\, f(x_{i_1,\dots,i_d}, m_{i_1,\dots,i_d})$$

$$\text{s.t.} \quad \mathcal{M} = [\![A_1, A_2, \dots, A_d]\!]$$

$m_1\times R \quad m_2\times R \quad m_d\times R$



We want to show

$$\frac{\partial F(\mathcal{M};\mathcal{X},\mathcal{W})}{\partial A_k} = \underset{m_k}{Y_{(k)}} \underset{R}{I} \quad \text{where } Y \in \mathbb{R}^{m_1\times\cdots\times m_d}$$

$1\times m_k\times R$

$$Y_{i_1,\dots,i_d} = W_{i_1,\dots,i_d}\, \frac{\partial f(x_{i_1,\dots,i_d}, M_{i_1,\dots,i_d})}{\partial M_{i_1,\dots,i_d}}$$

1×1

↳ of the same size as $\mathcal{X}$ but it is sparse if $\mathcal{W}$ is sparse.

$$= Y_{(k)} \left(A_d \odot \cdots \odot A_{k+1} \odot A_{k-1} \odot \cdots \odot A_1\right)$$

$$\frac{\partial F(M; X, W)}{\partial A_k} = \overbrace{\frac{\partial F(M; X, \Pi)}{\partial M}}^{y} \frac{\partial M}{\partial A_k}$$

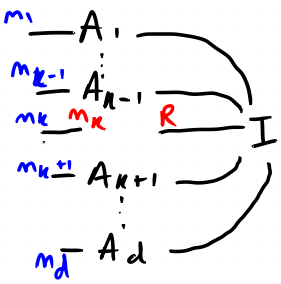
$I_X \quad m_k \times R \qquad\qquad 1 \times m_1 \times \cdots \times m_d \qquad m_1 \times \cdots \times m_d \times m_k \times R$

$$\frac{\partial F(M; X, \Pi)}{\partial M}_{i_1 \cdots i_d} = \partial \left(\sum_{j_1 \cdots j_d} W_{j_1 \cdots j_d} f(X_{j_1 \cdots j_d}, M_{j_1 \cdots j_d}) \right) / \partial M_{i_1 \cdots i_d}$$

$$= W_{i_1 \cdots i_d} \frac{\partial f(X_{i_1 \cdots i_d}, M_{i_1 \cdots i_d})}{\partial M_{i_1 \cdots i_d}} := y_{i_1 \cdots i_d}$$

$$\frac{\partial M}{\partial A_k} = \partial \left(\begin{matrix} -A_1 \\ -A_2 \\ \vdots \\ -A_d \end{matrix} \; I \right) / \partial A_k =$$

$m_1 \times \cdots \times m_d \times m_k \times R$



$$\frac{\partial F(M; X, W)}{\partial A_k} = \frac{\partial F(M; X, \Pi)}{\partial M} \frac{\partial M}{\partial A_k}$$

$$= \quad y \quad I \quad = y_{(k)} (A_d \odot \cdots \odot A_{k+1} \odot A_{k-1} \odot \cdots \odot A_1)$$

m_k R A_1 A_{k-1} A_{k+1} A_d

Follow up paper:

Stochastic Gradients for Large-Scale Tensor Decomposition*

Tamara G. Kolda† and David Hong‡